Solving Query-Answering Problems for the Semantic Web Using Equivalent Transformation

Kiyoshi Akama, Ekawit Nantajeewarawat, and Hidekatsu Koike

Abstract—A query-answering problem (QA problem) is an “all-answers finding” problem concerning with finding all ground instances of a query atomic formula that are logical consequences of a given logical formula. Based on the equivalent transformation (ET) principle, we propose a general framework for solving QA problems on Description Logics, which are logical formalisms underlying the Ontology Web Language OWL-DL. To solve such a QA problem using this framework, axioms and assertions representing the background knowledge of the problem are converted into a conjunction of first-order formulas, which is further converted by meaning-preserving Skolemization into a set of extended clauses typically containing global existential quantifications of function variables. The obtained clause set is then transformed successively using ET rules until the answer set of the original problem can be readily derived. ET rules for unfolding extended clauses, for removing useless extended clauses, and for dealing with function variables are presented. Application of the framework is illustrated.

Index Terms—Equivalent transformation, query answering problems, unfolding, skolemization, extended clause.

I. INTRODUCTION

Tableau-based algorithms [1], [2] have been successfully used in Description Logics (DLs) for checking the consistency of a knowledge base. For propositionally closed DLs (i.e., DLs with full Boolean operators), satisfiability checking and proof problems such as subsumption checking can be reduced to consistency checking [3], and can therefore be solved using tableau-based algorithms. These algorithms are invented specifically for dealing with axioms and assertions in DLs without converting them into equivalent formulas in first-order logic.

Query-answering problems (QA problems) are an important class of problems in the Semantic Web. Given a logical formula K, representing background knowledge, and an atomic formula (atom) a, representing a query, a QA problem is to find the set of all ground instances of a that are logical consequences of K. It is characteristically an “all-answers finding” problem, i.e., all ground instances of the query atom satisfying the requirement must be found. This is in contrast with a “yes/no” problem such as a proof problem, which is concerned with checking whether or not one given logical formula is a logical consequence of another given logical formula.

Most studies on solving proof problems and QA problems on DLs can be characterized as specific approaches, since they restrict logical formulas under consideration to certain specific subclasses of first-order logic, with a belief that more efficient computation can be achieved by such restriction. For solving QA problems, however, these specific approaches have not been so successful. For instance, solving a QA problem on DLs using proof methods, such as tableau-based algorithms, typically involves generating all ground instances of a query atom and then checking each generated instance individually whether it is a logical consequence of the background knowledge in use. Such a generate-and-check method is usually inefficient.

By contrast, we take a general approach to dealing with QA problems. In our approach, the background knowledge of a QA problem is converted into a first-order formula and a method is developed for solving QA problems on full first-order logic by means of equivalent transformation (ET). Taking this approach, the following fundamental issues need to be addressed:

1) Conversion of a first-order formula to a clausal form involves Skolemization. How to preserve the meaning of a first-order formula in a Skolemization process? Conventional Skolemization does not preserve the meaning of a formula [4]. In order to obtain meaning-preserving Skolemization, an extended formula space that allows existential quantifications of function variables is required.

2) How to achieve meaning-preserving transformation of an extended clause set? A clause set in the extended space contains existentially quantified global function variables. How to equivalently transform such an extended clause set has not been discussed in the literature.

A solution to the first issue has recently been provided by [5], in which a theory for extending a space of first-order formulas by incorporation of function variables was developed and how meaning-preserving Skolemization could be achieved in the obtained space of extended clauses, called the ECLS<sub>F</sub> space, was shown. A procedure for converting a first-order formula into a set of extended clauses in the ECLS<sub>F</sub> space was also given in [5].

The ET principle enables the use of various kinds of ET
rules for problem transformation. To address the second issue, ET rules for transforming extended clauses in the ECLS$_F$ space can be invented. With meaning-preserving Skolemization and ET-based problem solving, our approach deals with a QA problem as follows: First, convert a logical formula representing background knowledge into a first-order formula. Secondly, convert the obtained formula into a set of extended clauses in the ECLS$_F$ space using meaning-preserving Skolemization. Next, apply ET rules for problem transformation in the ECLS$_F$ space. An ET rule for unfolding extended clauses (possibly with more than one positive literal), an ET rule for removal of useless definite clauses, ET rules for dealing with atoms with function variables, and an ET rule for removal of subsumed clauses are demonstrated in this paper.

To begin with, Section II formalizes QA problems as model intersection problems and describes a general procedure for solving them using ET. Section III recalls the relationship between DLs and first-order logic. Section IV introduces meaning-preserving Skolemization, extended clauses in the ECLS$_F$ space, and QA problems on ECLS$_F$. Section V presents ET rules for unfolding and definite-clause removal in the ECLS$_F$ space. Section VI gives some additional ET rules. Section VII illustrates application of the presented ET-based procedure. Section VIII provides conclusions.

II. SOLVING QUERY-ANSWERING PROBLEMS USING EQUIVALENT TRANSFORMATION

A. Query-Answering (QA) Problems

A query-answering problem (QA problem) is a pair $\langle K, a \rangle$, where $K$ is a logical formula, representing background knowledge, and $a$ is an atomic formula (atom), representing a query. The answer to a QA problem $\langle K, a \rangle$, denoted by $\text{ansSet}(K, a)$, is defined as the set of all ground instances of $a$ that are logical consequences of $K$. This set can also be equivalently defined by

$$\text{ansSet}(K, a) = (\cap \text{Models}(K)) \cap \text{rep}(a),$$

where $\text{Models}(K)$ denotes the set of all models of $K$ and $\text{rep}(a)$ the set of all ground instances of $a$. When $K$ consists of only definite clauses, problems in this class are problems that have been discussed in logic programming [6]. In the class of QA problems discussed in [7], $K$ is a conjunction of axioms and assertions in Description Logics [3]. Answering queries in Datalog and deductive databases [8] can be regarded as solving QA problems on a restricted form of definite clauses.

Recently, QA problems have gained wider attention, owing partly to emerging applications in systems involving integration between formal ontological background knowledge and instance-level rule-oriented components, e.g., interaction between Description Logics and Horn rules [9], [10] in the Semantic Web’s ontology-based rule layer.

B. Solving QA Problems Using ET

To describe our procedure for solving QA problems using equivalent transformation (ET), the following notation is used. Assume that 1) $A$ is the set of all usual atoms, 2) $A_1$ and $A_2$ are disjoint subsets of $A$, and 3) $\phi$ is a bijection from $A_1$ to $A_2$. For any $i, j \in \{1, 2\}$, an extended clause in the ECLS$_F$ space is said to be from $A_i$ to $A_j$ iff all usual atoms in its right-hand side belong to $A_i$ and all those in its left-hand side belong to $A_j$.

Let $\langle K, a \rangle$ be a QA problem such that $K$ is a first-order formula in which all usual atoms belong to $A_1$. To solve this problem using ET, $K$ is transformed by meaning-preserving transformation into a set $Cs$ of extended clauses from $A_1$ to $A_1$ in the ECLS$_F$ space and a singleton set $Q$ consisting only of the clause $(\phi(a) \leftarrow a)$ from $A_1$ to $A_2$ is constructed from the query atom $a$. The resulting QA problem $\langle Cs \cup Q, \phi(a) \rangle$ is then successively transformed using ET rules.

In order to make a clear separation between a set of extended clauses from $A_1$ to $A_1$ and a set of those from $A_1$ to $A_2$ in a transformation process of QA problems, the following notation is introduced: Given a set $Cs$ of extended clauses from $A_1$ to $A_2$, and an atom $b$ in $A_2$, let the triple $\langle Cs, Q, b \rangle$ denote the QA problem $\langle Cs \cup Q, b \rangle$. A QA problem $\langle Cs, Q, b \rangle$ can be transformed by changing $Cs$, by changing $Q$, or by changing both $Cs$ and $Q$.

Definition 1. A transformation of a QA problem $\langle Cs, Q, b \rangle$ into a QA problem $\langle Cs', Q', b' \rangle$ is equivalent transformation (ET) iff $\text{ansSet}(Cs \cup Q, b) = \text{ansSet}(Cs' \cup Q', b')$.

Let a QA problem $\langle K, a \rangle$ be given, where $K$ is a first-order formula in which all usual atoms belong to $A_1$. A procedure for solving the QA problem $\langle K, a \rangle$ using ET is described below:

1) Transform $K$ by meaning-preserving Skolemization into a clause set $Cs$ in the ECLS$_F$ space (see Section IV).

2) Successively transform the QA problem $\langle Cs, \{\phi(a) \leftarrow a\}, \phi(a) \rangle$ in the ECLS$_F$ space using unfolding and other ET rules (see Sections V and VI).

3) If a QA problem $\langle Cs', Q, \phi(a) \rangle$ is obtained such that $\text{Models}(Cs') = \emptyset$, then $\text{ansSet}(K, a) = \text{rep}(a)$.

4) If a QA problem $\langle Cs', F, \phi(a) \rangle$ is obtained such that $\text{Models}(Cs') \neq \emptyset$, and $F$ is a set of unit clauses from $A_1$ to $A_2$ such that the head of each clause in $F$ is an instance of $\phi(a)$, then $\text{ansSet}(K, a) = \phi^{-1}(\cup(\text{rep}(\text{head}(C)) \mid C \in F))$.

III. FROM DESCRIPTION LOGICS TO FIRST-ORDER LOGIC

Ontologies provide the Semantic Web with a common understanding of the basic semantic concepts used to annotate Web pages and high quality ontologies are crucial for the Semantic Web. Descriptions Logics (DLs) [3] have been widely used as tools for constructing, integrating, and maintaining ontologies. In particular, the W3C recommendation Ontology Web Language OWL-DL is a syntactic variant of the SHOIN(D) Description Logic.

The meaning of a DL language is usually given in a Tarski-style model-theoretic way. Alternatively, it can be given by a translation into first-order logic, where classes correspond to unary predicates, properties correspond to...
binary predicates, and subclass/subproperty axioms correspond to implications [11], [12]. More precisely, individuals are equivalent to constants, class expressions are equivalent to first-order formulas with one free variable, and property expressions are equivalent to first-order formulas with two free variables. An inclusion axiom is equivalent to a first-order formula consisting of an implication between two formulas with a free variable universally quantified at its outer level.

IV. MEANING-PRESERVING SKOLEMIZATION

By meaning-preserving Skolemization [5], a first-order formula $K$ is converted into a clause set in an extended clause space, called the ECLS$_F$ space. An extended clause in this space takes the form

$$a_1, \ldots, a_m \leftarrow b_1, \ldots, b_p, f_1, \ldots, f_r,$$

where each of $a_1, \ldots, a_m$, $b_1, \ldots, b_p$ is a usual atom or a constraint atom, and $f_1, \ldots, f_r$ are func-atoms, which are introduced as follows: Given any $n$-ary function constant or $n$-ary function variable $f$, an expression $\text{func} (f, t_1, \ldots, t_n, t_{n+1})$, where the $t_i$ are usual terms, is considered as an atom of a new type, called a func-atom. When $f$ is a function constant and the $t_i$ are all ground, the truth value of this atom is evaluated to be true iff $f(t_1, \ldots, t_n) = t_{n+1}$.

Referring to the extended clause above as $C$, the sets $\{a_1, \ldots, a_m\}$ and $\{b_1, \ldots, b_p, f_1, \ldots, f_r\}$ are called the left-hand side and the right-hand side, respectively, of $C$, denoted by $\text{lhs}(C)$ and $\text{rhs}(C)$, respectively. If $m = 1$, then $C$ is called an extended definite clause, the only atom in $\text{lhs}(C)$ is called the head of $C$, denoted by $\text{head}(C)$, and the set $\text{rhs}(C)$ is also called the body of $C$, denoted by $\text{body}(C)$. When no confusion is caused, an extended clause and an extended definite clause are also called a clause and a definite clause, respectively.

The clause set obtained by meaning-preserving Skolemization may be further transformed equivalently in the ECLS$_F$ space into another clause set for problem simplification. Unfolding and other transformation rules may be used. A QA problem $(\mathcal{C}, \mathcal{a})$ such that $\mathcal{C}$ is a clause set in the ECLS$_F$ space and $\mathcal{a}$ is a usual atom is called a QA problem on ECLS$_F$.

V. EQUIVALENT TRANSFORMATION ON ECLS$_F$

A. Unfolding Operation on ECLS$_F$

Assume that:
1) $\mathcal{C}$ is a clause set in the ECLS$_F$ space.
2) $D$ is a definite-clause set in the ECLS$_F$ space.
3) $\text{occ}$ is an occurrence of an atom $b$ in the right-hand side of a clause $C$ in $\mathcal{C}$.

By unfolding $\mathcal{C}$ using $D$ at $\text{occ}$, $\mathcal{C}$ is transformed into

$$(\mathcal{C} - \{C\}) \cup (\cup \{\text{resolvent}(C, C', b) \mid C' \in D\}),$$

where for each $C' \in D$, $\text{resolvent}(C, C', b)$ is defined as follows, assuming that $\rho$ is a renaming substitution for usual variables such that $C$ and $C'\rho$ have no usual variable in common:

- If $b$ and $\text{head}(C'\rho)$ are not unifiable, then $\text{resolvent}(C, C', b) = \emptyset$.
- If they are unifiable, then $\text{resolvent}(C, C', b) = \{C'\}$, where $C'$ is the clause obtained from $C$ and $C'\rho$ as follows, assuming that $\emptyset$ is the most general unifier of $b$ and $\text{head}(C')$:
  a) $\text{lhs}(C') = \text{lhs}(C)$
  b) $\text{rhs}(C') = (\text{rhs}(C) - \{b\}) \cup \text{body}(C'\rho \emptyset)$

The resulting clause set is denoted by $\text{UNFOLD}(\mathcal{C}, D, \text{occ})$.

B. ET by Unfolding and Definite-Clause Removal

For any predicate $p$, let $\text{Atoms}(p)$ denote the set of all atoms having the predicate $p$. Equivalent transformation of QA problems on ECLS$_F$ using unfolding and using definite-clause removal are formulated below.

Theorem 1. Let $(\mathcal{C}, \mathcal{a})$ be a QA problem on ECLS$_F$. Assume that:
1) $q$ is the predicate of the query atom $\mathcal{a}$.
2) $p$ is a predicate such that $p \neq q$.
3) $D$ is a set of definite clauses in $\mathcal{C}$ that satisfies the following conditions:
   a) For any definite clause $C \in D$, $\text{head}(C) \in \text{Atoms}(p)$.
   b) For any clause $C' \in \mathcal{C} - D$, $\text{lhs}(C') \cap \text{Atoms}(p) = \emptyset$.
4) $\text{occ}$ is an occurrence of an atom in $\text{Atoms}(p)$ in the right-hand side of a clause in $\mathcal{C} - D$.

Then the following two sets are equal:

- $(\bigcap \text{Models}(\mathcal{C})) \cap \text{rep}(a)$
- $(\bigcap \text{Models}(\text{UNFOLD}(\mathcal{C}, D, \text{occ}))) \cap \text{rep}(a)$

By Definition 1, a transformation given by Theorem 1 is ET.

Theorem 2. Let $(\mathcal{C}, \mathcal{a})$ be a QA problem on ECLS$_F$. Assume that:
1) $q$ is the predicate of the query atom $\mathcal{a}$.
2) $p$ is a predicate such that $p \neq q$.
3) $D$ is a set of definite clauses in $\mathcal{C}$ that satisfies the following conditions:
   a) For any definite clause $C \in D$, $\text{head}(C) \in \text{Atoms}(p)$.
   b) For any clause $C' \in \mathcal{C} - D$, $\text{lhs}(C') \cap \text{Atoms}(p) = \emptyset$.
4) $\text{occ}$ is an occurrence of an atom in $\text{Atoms}(p)$ in the right-hand side of a clause in $\mathcal{C} - D$.

Then the following two sets are equal:

- $(\bigcap \text{Models}(\mathcal{C})) \cap \text{rep}(a)$
- $(\bigcap \text{Models}(\mathcal{C} - D)) \cap \text{rep}(a)$

By Definition 1, a transformation given by Theorem 2 is ET.

VI. OTHER ET RULES ON ECLS$_F$

A. Merging Two Func-Atoms with the Same Call Pattern

An ET rule for merging func-atoms having the same invocation pattern is given below:

Proposition 1. Let $\mathcal{C}$ be a clause set in the ECLS$_F$ space. Assume that:
1) $f_1$ and $f_2$ are func-atoms that differ only in their last arguments.
2) $C$ is a clause such that $\{f_1, f_2\} \subseteq \text{rhs}(C)$.

Then:
- If the last arguments of \( f_1 \) and \( f_2 \) are unifiable, with their most general unifier being \( \theta \), and \( C \) is a clause such that
  a) \( \text{lhs}(C) = \text{lhs}(\theta) \), and
  b) \( \text{rhs}(C) = (\text{rhs}(C) - \{f_1, f_2\}) \cup \{f_1\theta\} \),
then \( \text{Models}(C \cup \{C\}) = \text{Models}(C) \cup \{C\} \).
- If their last arguments are not unifiable, then \( \text{Models}(C \cup \{C\}) = \text{Models}(C) \).

By Definition 1, a transformation using Proposition 1 is ET.

B. Elimination of Isolated Func-Atoms

Next, an ET rule for removing isolated func-atoms is given (Proposition 2). A func-atom \( \text{func}(h, t_1, ..., t_n, v) \), where \( v \) is a usual variable, is said to be isolated in a clause \( C \) iff the following conditions are satisfied:
1. There is only one occurrence of \( v \) in \( C \).
2. There is no func-atom \( \text{func}(h', t'_1, ..., t'_n, v') \) in \( C \) such that \( v' \) is a usual variable and \( [t_1, ..., t_n] \) and \( [t'_1, ..., t'_n] \) are unifiable.

Proposition 2. Let \( C \) be a clause set in the ECLS\(_F\) space. Assume that:
1) \( C \) is a clause such that \( C \) contains a func-atom that is isolated in \( C \).
2) \( C' \) is the clause obtained from \( C \) by removing all func-atoms that are isolated in \( C \).

Then \( \text{Models}(C \cup \{C\}) = \text{Models}(C \cup \{C'\}) \).

By Definition 1, a transformation using Proposition 2 is ET.

C. Elimination of Clauses by Subsumption

A clause \( C_1 \) is said to subsume a clause \( C_2 \) iff there exists a substitution \( \theta \) for usual variables such that \( \text{lhs}(C_1) \theta \subseteq \text{lhs}(C_2) \) and \( \text{rhs}(C_1) \theta \subseteq \text{rhs}(C_2) \). A subsumed clause can be removed by the following ET rule:

Proposition 3. Let \( C \) be a clause set in the ECLS\(_F\) space. For any clauses \( C_1 \) and \( C_2 \), if \( C_1 \) subsumes \( C_2 \), then \( \text{Models}(C \cup \{C, C_2\}) = \text{Models}(C \cup \{C_1\}) \).

By Definition 1, a transformation using Proposition 3 is ET.

VII. Example

To illustrate application of the procedure in Section II-B and ET rules in Section V and Section VI, consider the “Tax-cut” problem discussed in [10]. This problem is to find all persons who can have discounted tax, with the knowledge that 1) any person who has two children or more can get discounted tax, 2) men and women are not the same, 3) a person’s mother is always a woman, 4) Peter has a child named Paul,5) Paul is a man, and 6) Peter has a child, who is someone’s mother.

This background knowledge is represented in the SHOIN(D) Description Logic, which is the logical formalism underlying the Web Ontology Language OWL-DL, as the following axioms and assertions:

\[ \exists \text{2 hasChild} \subseteq \text{TaxCut} \]
\[ \text{Man} \cap \text{Woman} \subseteq \bot \]
\[ \exists \text{motherOf.Person} \subseteq \text{Woman} \]

Theses axioms and assertions are translated into the following first-order formulas, assuming that all individuals under consideration are instances of Person and that hasChild, TaxCut, Man, Woman, and motherOf are abbreviated, respectively, to hc, tc, mn, wm, and mo:

\[ F_1: \forall x: ((\exists y_1 \exists y_2:\ (hc(x, y_1) \land hc(x, y_2) \land notSame(y_1, y_2)) \rightarrow tc(x)) \]
\[ F_2: \forall x \forall y: ((mn(x) \land wm(y)) \rightarrow notSame(x, y)) \]
\[ F_3: \forall x: ((\exists y: mo(x, y)) \rightarrow wm(x)) \]
\[ F_4: hc(Peter, Paul) \]
\[ F_5: mn(Paul) \]
\[ F_6: \exists x: (hc(Peter, x) \land (\exists y: mo(x, y))) \]

Accordingly, the “Tax-cut” problem can be formulated as a QA problem \((K, tc(x))\) on first-order logic, where \( K \) is the conjunction of the above six first-order formulas. Using the meaning-preserving Skolemization procedure given in [5], the first-order formula \( K \) is transformed into a clause set \( C \) consisting of the following extended clauses:

\[ C_1: tc(x) \leftarrow hc(x, y_1), hc(x, y_2), notSame(y_1, y_2) \]
\[ C_2: notSame(y, x) \leftarrow mn(x), wm(y) \]
\[ C_3: wm(x) \leftarrow mo(x, y) \]
\[ C_4: hc(Peter, Paul) \leftarrow \]
\[ C_5: mn(Paul) \leftarrow \]
\[ C_6: hc(Peter, x) \leftarrow \text{func}(h_1, x) \]
\[ C_7: mo(x, y) \leftarrow \text{func}(h_1, x), \text{func}(h_2, y) \]

The clauses \( C_6 \) and \( C_7 \) together represent the first-order formula \( F_6 \), where \( h_1 \) and \( h_2 \) are 0-ary function variables. Assume that all usual atoms occurring in \( C \) belong to \( A_1 \) and \( \phi(tc(x)) = ans(x) \). Let \( C_0 \) be given by:

\[ C_0: ans(x) \leftarrow tc(x) \]

To solve the QA problem \((K, tc(x))\), the QA problem \((C, \{C_0\}, ans(x))\) is successively transformed by applying the ET rules in Sections V and VI as follows:
1) By unfolding \( C_0 \) at \( tc(x) \) using \( \{C_1\} \), \( C_0 \) is replaced with:

\[ C_8: ans(x) \leftarrow hc(x, y_1), hc(x, y_2), notSame(y_1, y_2) \]
2) \( C_1 \) is removed.
3) By unfolding \( C_8 \) at the last body atom using \( \{C_2\} \), \( C_8 \) is replaced with:

\[ C_9: ans(x) \leftarrow hc(x, y_1), hc(x, y_2), mn(y_1), wm(y_2) \]
4) \( C_3 \) is removed.
5) By unfolding \( C_9 \) at the 3rd body atom using \( \{C_5\} \), \( C_9 \) is replaced with:

\[ C_{10}: ans(x) \leftarrow hc(x, Paul), hc(x, y_2), wm(y_2) \]
6) \( C_5 \) is removed.
7) By unfolding \( C_{10} \) at the 3rd body atom using \( \{C_6\} \), \( C_{10} \) is replaced with:

\[ C_{11}: ans(x) \leftarrow hc(x, Paul), hc(x, y_2), mo(y_2, z) \]
8) C₃ is removed.
9) By unfolding C₁₁ at the 3rd body atom using \{C₇\}, C₁₁ is replaced with:

\[
C_{11}: \text{ans}(x) \leftarrow \text{hc}(x, \text{Paul}), \text{hc}(x, y₂), \text{func}(h₁, y₂), \text{func}(h₂, z)
\]

10) C₇ is removed.
11) By removing an isolated func-atom, C₁₂ is replaced with:

\[
C_{12}: \text{ans}(x) \leftarrow \text{hc}(x, \text{Paul}), \text{hc}(x, y₂), \text{func}(h₁, y₂)
\]

12) By unfolding C₁₃ at the 1st body atom using \{C₄, C₆\}, C₁₃ is replaced with:

\[
C_{13}: \text{ans}(x) \leftarrow \text{hc}(x, \text{Paul}), \text{hc}(x, y₂), \text{func}(h₁, y₂)
\]

13) By merging func-atoms with the same invocation pattern, C₁₅ is replaced with:

\[
C_{15}: \text{ans}(\text{Peter}) \leftarrow \text{hc}(x, \text{Paul}), \text{hc}(\text{Peter}, y₂), \text{func}(h₁, y₂)
\]

14) Since C₁₅ is subsumed by C₁₆, C₁₆ is removed.
15) By unfolding C₁₄ at the 1st body atom using \{C₄, C₆\}, C₁₄ is replaced with:

\[
C_{14}: \text{ans}(\text{Peter}) \leftarrow \text{hc}(x, \text{Paul}), \text{hc}(\text{Peter}, y₂), \text{func}(h₁, y₂)
\]

16) C₄ and C₆ are removed.
17) By merging func-atoms with the same invocation pattern, C₁₈ is replaced with:

\[
C_{18}: \text{ans}(\text{Peter}) \leftarrow \text{func}(h₁, y₂)
\]

18) By removing an isolated func-atom, C₁₉ is replaced with:

\[
C_{20}: \text{ans}(\text{Peter}) \leftarrow
\]

19) Since C₁₇ is subsumed by C₂₀, C₁₇ is removed.

The resulting QA problem is (\emptyset, \{C₂₀\}, \text{ans}(x)). Since Models(\emptyset) \neq \emptyset and C₂₀ is a unit clause, the answer to the “Tax-cut” problem (K, tc(x)) is determined by

\[
\phi^{-1}(\cup\{\text{rep}(\text{head}(C₂₀))\}) = \phi^{-1}(\{\text{ans}(\text{Peter})\}) = \{\text{tc}(\text{Peter})\},
\]

i.e., Peter is the only person who gets discounted tax.

VIII. CONCLUSIONS

We have proposed a general approach to solving QA problems for the Semantic Web, using ET-based problem solving. This approach consists of three main phases: 1) conversion of the background knowledge of a given QA problem into a first-order formula, 2) conversion of the resulting first-order formula through meaningful Preserving Skolemization in the extended formula space ECLSₚ, and 3) reduction of problems by application of ET rules on ECLSₚ. The presented framework allows one to deal with QA problems on a large class of logical formulas, including an extension of Description Logics by incorporation of declarative rules, which offer more extensive facilities for representing and reasoning with individuals and relations between them. QA problems on such an extended representation formalism are an important class of problems in the Semantic Web’s ontology-based rule layer.

REFERENCES


Kiyoshi Akama received his DEng from Tokyo Institute of Technology. He is a professor at Division of Large-Scale Computational Systems, Information Initiative Center, Hokkaido University. His research interests include program generation and computation based on the equivalent transformation model, programming paradigms, knowledge representation, semantic web, and e-learning. He is a member of the Japanese Society for Artificial Intelligence and the Information Processing Society of Japan.

Ekawit Nantajeewarawat received his DEng in Computer Science from the Asian Institute of Technology. He is an associate professor at Computer Science Program, Sirindhorn International Institute of Technology, Thammasat University. His research interests include knowledge representation and automated reasoning. He is a member of the Association for Computing Machinery and the Institute of Electronics, Information and Communication Engineers.

Hidekatsu Koike received his PhD from Systems and Information Engineering, Kyushu University. He is an associate professor at the Faculty of Social Information, Sapporo Gakuen University. His research interests include program generation and computation based on equivalent transformation. He is a member of the Association for Computing Machinery, IEEE Computer Society and the Institute of Electronics, Information and Communication Engineers.